Szegedi SZC
Tóth János Mórahalmi Szakképző Iskola és Szilágyi Mihály Kollégium

ZÁRÓDOLGOZAT

PONG JÁTÉK FEJLESZTÉSE

Tanuló

BOZÓKI MÁTÉ

Mentor

SZABÓ BEATRIX

Mórahalom, 2023

Tartalom

Bevezetés

Az asztali alkalmazások napjainkban egyre nagyobb népszerűségnek örvendenek, és számos területen megtalálhatók. Ezek az alkalmazások számos lehetőséget kínálnak szórakozásra, munkára vagy akár tanulásra is. Ebben a dolgozatban az ikonikus Pong játék asztali alkalmazását mutatom be.

A Pong egy olyan videojáték, amelyet az Atari fejlesztett ki és adott ki 1972-ben. Az eredeti játék célja egyszerű volt: két játékos egy-egy asztali teniszütőt irányított a képernyőn, és a labdát kellett visszapattintaniuk egymás felé, hogy megakadályozzák annak átjutását a saját oldalukra. A játék gyorsan hatalmas népszerűségre tett szert, és a videojáték-ipar egyik első jelentős sikere lett.

Az asztali alkalmazás verziója lehetővé teszi, hogy a játékot közvetlenül a számítógépen élvezd. Az alkalmazásban számos testre szabási lehetőség áll rendelkezésedre, mint például a játék sebességének beállítása, a pálya kialakítása és még sok más. Emellett lehetőséged van egyedül játszani a számítógép ellen, vagy akár barátaiddal is játszhatsz.

Ebben a dolgozatban részletesen bemutatom az asztali alkalmazás Pong játékának felépítését, funkcionalitását és tervezési döntéseit. Áttekintést adok a játék fejlesztési folyamatáról, a felhasznált technológiákról és az esetleges kihívásokról. Ezen túlmenően bemutatom a játékot tesztelésének eredményeit, valamint az alkalmazás továbbfejlesztését.

Az asztali alkalmazás Pong játéka izgalmas és szórakoztató élményt nyújt mind a játékosoknak, mind a fejlesztőknek. Lássuk, hogyan valósítottam meg ezt a klasszikus játékot a digitális világban!

A Pong mára már kultikus népszerűségű lett, tévéműsorokban is utalásokat tettek rá, valamint más videójátékok is."[1] A célközönség a 10-14 éves korú gyerekek.

[1] Wikipédia - Pong játék

1. Fejlesztői dokumentáció

1.1. A projekt céljának ismertetése

A célom az volt, hogy készítsek egy egyszerű Pong játékot HTML,CSS,JavaScript nyelven. Pong egy klasszikus asztali játék, amelyet általában két játékos játszik. A játék célja az, hogy a játékosok ütőket használva elütögessék a labdát a pályán és megpróbálják az ellenfél oldalfalához juttatni a labdát. Azért választottam a játékfejlesztés témakört a dolgozatom elkészítéséhez, mivel nagyon elnyerte a tetszésem a tananyag elsajátítása során.

A Pong játék fejlesztése során a célom az alábbiak voltak:

- Fejleszteni a programozási készségeket és megérteni az objektumorientált programozás alapjait.
- Elmélyíteni a grafikus felhasználói felületek, például az ablakok, gombok és vezérlők kezelésében.
- Gyakorolni az esemény vezérelt programozást
- Megérteni a játékfejlesztés alapjait, például a mozgás, a gravitáció és a kollúzió kezelését.
- Fejleszteni az algoritmusok és az adatszerkezetek megértését.
- Tanulni a játék tervezésének és implementálásának alapjait.

A program célja, hogy szórakoztató legyen az emberek számára és kikapcsolódást biztosítson.

A következő funkciókat tartalmazza:

1. Játéktér és elemek inicializálása:
    - Létrehoztam egy játéktér ablakot, amelyben a játék zajlik.
    - Létrehoztam két ütőt (játékosok), amelyek a pályán mozogni fognak.
    - Létrehoztam egy labdát, amely a játékteret fogja átszelni.
2. Ütközések kezelése:
    - Meghatároztam a játéktér határait, ahol a labda ütközhet.
    - Kezeltem az ütők és a labda ütközéseit, hogy megfelelő irányban változzon a labda mozgása.
3. Játékosok mozgatása:
    - Megadtamaz ütők mozgását (fel, le mozgatás).
    - Korlátoztam a játékosok mozgását, hogy ne léphessenek ki a játéktérből.
4. Játéklogika:
    - A játék futása közben ellenőriztem a játékosok és a labda pozícióját.

- Ellenőriztem, hogy a labda elérte-e a játékteret határoló falakat, vagy az egyik játékos ütőjét.
- Frissítettem a játék állapotát, azaz pontszámokat, sebességet, stb.

5. Pontozás:

- Nyomon követtem a játékosok pontjait.
- Amikor a labda az egyik játékos oldalának a mögé ér, növeltem az ellenfél pontszámát.

6. Játék vége:

- Ellenőriztem a pontszámokat, hogy meghatározzam, melyik játékos nyert.
- Ha egy játékos elért egy meghatározott pontszámot, a játék véget ér és kiírja az eredményt.

1.2. Telepítési útmutató

A játékot a következő github linkről töltheti le:

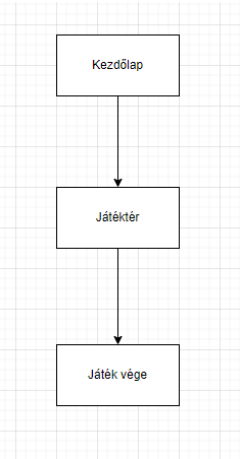
https://github.com/Szoftverfejlesztok/Szakmai_vizsga_Bozoki_Mate.git

A letöltést követően ki kell csomagolni a projektet és az index.html fájl megnyitásával elindul a játék.

1.3. Az alkalmazás bemutatása

Az alkalmazás három ablakból épül fel. Ezek az ablakok a következők:

1. Kezdőképernyő
2. Játéktér
3. Játék vége



1.ábra - Rendszerterv

1.3.1. Az adatok tárolása

A játék csak a játékállapotot tárolja a memóriában, például a játékosok pontszámait és a labda pozícióját, de nem menti ezeket az adatokat adatbázisba vagy más tartós tárolási helyre. Amikor a játék újraindul vagy frissül, az állapot alaphelyzetbe kerül.

1.3.2. HTML fájl szerkezete

A pong_game.html egy egyszerű HTML dokumentumot tartalmaz, amely egy Ping Pong játék felhasználói felületét valósítja meg. Az alábbiakban található részletes magyarázat a kód különböző elemeire:

- Az <html> elemben megadott lang attribútum az oldal nyelvét határozza meg, ebben az esetben "en" (angol).
- A <head> elem tartalmazza a dokumentum fejlécéhez kapcsolódó információkat, például a karakterkódolást (<meta charset="UTF-8" />) és a nézeti ablak beállításait (<meta name="viewport" content="width=device-width, initial-scale=1.0" />).
- Az oldal címe a <title> elemen belül található, itt "Ping Pong Game" néven van megadva.
- A CSS stílusokat a <link> elemek segítségével hivatkozzák be. Az első link az oldal által használt stíluslapot (style.css) tartalmazza, a második link pedig az oldal ikonját (favicon.ico) adja meg.
- Az oldal tartalmát a <body> elemben található elemek határozzák meg.
- Az egyik fő elem a <canvas>, amely a játékterületet reprezentálja.
- A játékhoz tartozik egy felugró panel, amely a játékstatisztikákat és a játékmódokat tartalmazza. A panel a elemmel van megadva, aminek az osztálya "panel reveal". A panel tartalmazza a következő elemeket:
- Egy elem a játékstatisztikák megjelenítéséhez. Az osztálya "stat".
- Egy <h1> cím az oldal fejlécében, amely a "Ping Pong Game" szöveget tartalmazza.
- Egy elem a játékmódok szöveges leírásához. A szöveg mellett található egy elem, aminek az osztálya "tooltip" és tartalmazza a "A mode sharpens the head of computer. not fastens the game!" szöveget.
- Egy rejtett <input> elem típusa "text". Ez a mező valószínűleg a játék beállításainak tárolására szolgál, de az itt látható kódrészletben nincs további információ a céljáról.
- Egy <ul> elem osztálya "gameMode". Ez egy listaelemeket tartalmazó lista, ahol mindegyik <li> elem egy játékmódot reprezentál. Minden <li> elemnek van egy osztálya, például

"baby", "easy", "normal", stb., és egy adatattribútuma, amely a játék nehézségi szintjét határozza meg (például "data-number=".05"").

- Egy <button> elem az "Play" felirattal, amelyet a felhasználó kattintással aktiválhat.
- Végül a <script> elem a resources/js/main.js fájlra hivatkozik, amely a játék logikáját és működését tartalmazza.

```
<html lang="en">
    <head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <meta http-equiv="X-UA-Compatible" content="IE=edge" />
        <meta name="keywords" content="Document" />
        <meta name="description" content="Your Description..." />
        <title>Ping Pong Game</title>
        <link rel="stylesheet" href="resources/css/style.css" />
        <link
            rel="shortcut icon"
            href="resources/img/favicon.ico"
            type="image/x-icon"
        />
    </head>
    <body>
        <canvas></canvas>
        <div class="panel reveal">
            <div class="stat"></div>
            <h1>Ping Pong Játék</h1>
            <span> Választj egy játék módot, majd kattints a játék kezdete gombra! <br> Játék módok:<div
class="tooltip"></div></span>
            <input type="text" hidden>
            <ul class="gameMode">
                <li class="mode baby" data-number=".05">Kezdő</li>
                <li class="mode easy" data-number=".08">Könnyű</li>
                <li class="mode normal" data-number=".1">Normál</li>
                <li class="mode hard" data-number=".2">Nehéz</li>
                <li class="mode insane" data-number=".3">Profi</li>
            </ul>
            <button class="play">Játék kezdete</button>
        </div>
        <script src="resources/js/main.js"></script>
    </body>
</html>
```

1.3.3. CSS fájl tartalma

Az alábbi kódrészlet a következő részekből áll:

Betűtípus importálása:

- Az @import szabály segítségével két betűtípust importálunk a Google Fonts-ból, a Montserrat és a Berkshire Swash fontokat.

HTML és test (body) elemek:

- A <html> elemhez és a testhez (body) néhány stílus tulajdonságot adtunk meg.
- A scroll-behavior: smooth; beállítás sima görgetést eredményez az oldalon.
- A testnek (body) nullára állítjuk a margót (margin) és a belső töltést (padding), valamint a dobozmodellt a border-box-ra állítjuk.
- A betűtípust a Montserrat-ra állítjuk be, a háttérképet pedig a bg.jpg képre állítjuk, amely középre igazított és teljesen kitölti a háttérteret.
- A háttérkép mérete a cover, így az az ablak méretéhez igazodik.
- A backdrop-filter: blur(10px); tulajdonsággal egy elmosott hatást adunk a háttérképnek.

- A test (body) szélességét 100%-ra és a magasságát 100vh-ra állítjuk be, hogy teljesen kitöltse a rendelkezésre álló teret.

Vászon (canvas) elem:

- A vászonhoz néhány stílus tulajdonságot adtunk meg.
- A position: absolute; beállítással a vászon a dokumentum legfelső bal sarkához igazodik.
- A top, left, bottom és right tulajdonságokkal középre igazítjuk a vásznat.
- A margin: auto; beállítással középre igazítjuk a vásznat.
- A border tulajdonsággal 10 képpont vastagságú, #e9f2ff színű keretet adunk a vászonnak.
- A border-radius tulajdonsággal 20 képpontos sugarat állítunk be a vászonhoz.

Panel:

- A panelhez néhány stílus tulajdonságot adtunk meg.
- A position: fixed; beállítással a panel rögzített pozícióban marad, ha a felhasználó görget az oldalon.
- A top: 50%; left: 50%; tulajdonságokkal középre igazítjuk a panelt.
- A max-width: 600px; beállítással maximális szélességet határozunk meg a panelnek.
- A height: 400px; beáll tással a panel magasságot 400 képpontra állítjuk be.
- A background: white; tulajdonsággal fehér háttérszínt adunk a panelnek.
- A border-radius: 30px; beállítással 30 képpontos sugarat határozunk meg a panel széleinek lekerekítéséhez.
- A border: 10px solid #e9f2ff; tulajdonsággal 10 képpont vastag, #e9f2ff színű keretet adunk a panelnek.
- A display: flex; beállítással a panelt flex konténerként kezeljük.
- Az justify-content: center; align-items: center; tulajdonságokkal középre igazítjuk a panel tartalmát.
- A flex-direction: column; beállítással a flex elemeket függőlegesen rendezzük el.
- A font-family: 'Berkshire Swash', cursive; tulajdonsággal a Berkshire Swash betűtípust állítjuk be a panel szövegeihez.
- A transform: translate(-50%, -50%) scale(0); tulajdonsággal a panel 50%-kal vízszintesen és függőlegesen eltolódik, és 0-s méretarányt kap.
- Az opacity: 0; tulajdonsággal az átláthatóságot 0-ra állítjuk.
- A z-index: -1000; beállítással a panel rétegrendjét -1000-re állítjuk, hogy háttérbe kerüljön.
- Az transition: .5s ease-in-out; tulajdonsággal animációt definiálunk a panelhez, 0,5 másodperces időtartammal és be- és kilassuló módosító függvénnyel.

Reveal class:

- A .reveal class-t a panelre alkalmazzuk, hogy bekapcsoljuk az animációt és megjelenítsük a panelt.
- A transform: translate(-50%, -50%) scale(1); tulajdonsággal a panel visszatér az eredeti pozíciójába és méretarányába.
- Az opacity: 1; tulajdonsággal az átláthatóságot 1-re állítjuk.
- A z-index: 1000; beállítással a panel rétegrendjét 1000-re állítjuk, hogy előtérbe kerüljön.

Egyéb stílusok:

- A .panel .stat, .panel h1, .panel span osztályokhoz néhány stílus tulajdonságot adtunk meg, például szöveg színt, betűméretet és margókat.
- A .panel span .tooltip osztályhoz néhány stílus tulajdonságot ad

```
@import
url('https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&display=swap');
@import url('https://fonts.googleapis.com/css2?family=Berkshire+Swash&display=swap');

html {
    scroll-behavior: smooth;
}

body {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: 'Montserrat', 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif;
    background: url("../img/bg.jpg") no-repeat center center;
    background-size: cover;
    backdrop-filter: blur(10px);
    width: 100%;
    height: 100vh;
}

canvas {
    position: absolute;
    top: 0;
    left: 0;
    bottom: 0;
    right: 0;
```

```
    margin: auto;
    border: 10px solid #e9f2ff;
    border-radius: 20px;
}

.panel {
    position: fixed;
    top: 50%;
    left: 50%;
    max-width: 600px;
    height: 400px;
    background: white;
    border-radius: 30px;
    border: 10px solid #e9f2ff;
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
    font-family: 'Berkshire Swash',
    cursive;
    transform: translate(-50%, -50%) scale(0);
    opacity: 0;
    z-index: -1000;
    transition: .5s ease-in-out;
}

.panel.reveal {
    transform: translate(-50%, -50%) scale(1);
    opacity: 1;
    z-index: 1000;
}

.panel .stat {
    color: #33485a;
    font-size: 50px;
    margin-top: 0;
}

.panel h1 {
    color: #142431;
    font-size: 20px;
    margin-top: 0;
}

.panel span {
    position: relative;
    font-size: 20px;
    text-align: center;
}

.panel span .tooltip {
    position: absolute;
    top: -100%;
    left: 50%;
    visibility: hidden;
    transform: translate(-50%, -50%);
    opacity: 0;
    width: 300px;
    background: #142431;
    color: #f2f4ff;
    border-radius: 10px;
    text-transform: capitalize;
    z-index: 2;
    transition: .3s;
}

.panel span:hover .tooltip {
    opacity: .6;
    visibility: visible;
}

.panel .gameMode {
    display: block;
    position: relative;
    text-align: center;
    padding: 0;
}

.panel .gameMode .mode {
    display: inline-block;
    list-style: none;
    font-weight: bold;
    font-size: 25px;
    padding: 10px 30px;
    border-radius: 50%;
```

```
    border: none;
    cursor: pointer;
    color: white;
    transition: .25s ease-in-out;
}

.panel .gameMode .mode:hover {
    transform: scale(1.2);
}

.panel .gameMode .mode.selected {
    border: 5px solid #fa7075;
}

.panel .gameMode .mode:nth-child(1) {
    background: linear-gradient(#96ff55, #609b1f);
}

.panel .gameMode .mode:nth-child(2) {
    background: linear-gradient(#49f514, #07aa08);
}

.panel .gameMode .mode:nth-child(3) {
    background: linear-gradient(#fa7a70, #be3b44);
}

.panel .gameMode .mode:nth-child(4) {
    background: linear-gradient(#f8884b, #f96618);
}

.panel .gameMode .mode:nth-child(5) {
    background: linear-gradient(rgb(255, 119, 69), red);
}

.play {
    display: inline-block;
    list-style: none;
    font-weight: bold;
    font-size: 25px;
    padding: 10px 100px;
    border-radius: 50%;
    cursor: pointer;
    color: white;
    outline: none;
    border: none;
    background: linear-gradient(#5403bf, #c93f9c);
    transition: .25s ease-in-out;
}

.play:hover {
    transform: scale(1.2);
}
```

1.2.3. A main.js fájl tartalma

Változók inicializálása:

- canvas: A játékterületet reprezentáló Canvas elem.
- ctx: A Canvas 2D kontextusa.
- panel: A panel elem a játékban.
- play: A játék indításáért felelős gomb.
- mode: Az input elem, amelyen keresztül a játékos kiválaszthatja a nehézségi szintet.
- modeValues: A nehézségi szinteket reprezentáló elemek listája.
- statPanel: A statisztikákat megjelenítő panel elem.
- comScore: Hangfájl az ellenfél pontszerzése esetén.
- userScore: Hangfájl a játékos pontszerzése esetén.
- user: Az játékos adatait tartalmazó objektum.
- computer: Az ellenfél adatait tartalmazó objektum.
- ball: A labda adatait tartalmazó objektum.
- net: A háló adatait tartalmazó objektum.
- Particle: Az osztály sablon a részecskék létrehozásához.

Rajzoláshoz szükséges függvények:

- drawRect: Téglalap rajzolása.
- drawArc: Kör rajzolása.
- drawText: Szöveg rajzolása.
- drawNet: Háló rajzolása.

Paddle és labda mozgatása:

- canvas eseményfigyelők: Az egér vagy érintőképernyő mozgatására reagálva változtatja a játékos paddle pozícióját.
- collision: Ütközésdetektáló függvény a paddle és a labda között.

Játék működése:

- resetBall: A labda pozíciójának alaphelyzetbe állítása.
- gameOver: Játék vége esetén végrehajtandó kód.
- draw: A játékterület kirajzolása.
- game: A játék fő ciklusa, amelyben a labda mozgása, ütközések kezelése és pontszámok frissítése történik.

Eseményfigyelők:

- play gombra kattintva elindul a játék.
- resize eseményre válaszolva változtatja a Canvas méretét.

```
// Elemek kiválasztása
const canvas = document.querySelector("canvas");
const ctx = canvas.getContext("2d");
const panel = document.querySelector(".panel");
const play = document.querySelector(".play");
const mode = document.querySelector("input[type=text]");
const modeValues = document.querySelectorAll(".mode");
const statPanel = document.querySelector(".stat");

const comScore = new Audio();
comScore.src = "resources/audio/comScore.mp3";
const userScore = new Audio();
userScore.src = "resources/audio/userScore.mp3";

// A felhasználó kiválasztja a játék módot.
modeValues.forEach((modeValue) => {
    modeValue.addEventListener("click", () => {
        modeValues.forEach((index) => {
            index.classList.remove("selected");
        });
        modeValue.classList.add("selected");
        let value = modeValue.dataset.number;
        mode.value = value;
    });
});
let compSpeed; // A számítógép sebessége, amely kiválasztja a mód adatkészletét.

canvas.width = 600;
canvas.height = 400;

let user = {
    width: 10,
    height: 100,
    color: "white",
    x: 10,
    y: (canvas.height - 100) / 2,
    score: 0,
};

let computer = {
    width: 10,
```

```
    height: 100,
    color: "white",
    x: canvas.width - 20,
    y: (canvas.height - 100) / 2,
    score: 0,
};

let ball = {
    radius: 10,
    velocity: {
        x: 7,
        y: 7,
    },
    speed: 7,
    color: "white",
    x: canvas.width / 2,
    y: canvas.height / 2,
};

let net = {
    x: (canvas.width - 2) / 2,
    y: 0,
    color: "white",
    width: 2,
    height: 10,
};

// Funkció téglalap, kör, szöveg, háló rajzolásához.
function drawRect(x, y, w, h, c) {
    ctx.beginPath();
    ctx.fillStyle = c;
    ctx.fillRect(x, y, w, h);
    ctx.closePath();
}

function drawArc(x, y, r, c) {
    ctx.beginPath();
    ctx.fillStyle = c;
    ctx.arc(x, y, r, 0, Math.PI * 2);
    ctx.fill();
    ctx.closePath();
}

function drawText(text, x, y) {
    ctx.font = "75px arial";
    ctx.fillStyle = "white";
    ctx.fillText(text, x, y);
}

function drawNet() {
    for (let i = 0; i < canvas.width; i += 15) {
        drawRect(net.x, net.y + i, net.width, net.height, net.color);
    }
}

class Particle {
    constructor(x, y, color, radius, velocity) {
        this.x = x;
        this.y = y;
        this.color = color;
        this.radius = radius;
        this.velocity = velocity;
    }
    draw() {
        ctx.beginPath();
        ctx.arc(this.x, this.y, Math.abs(this.radius), 0, Math.PI * 2, false);
```

```
        ctx.fillStyle = this.color;
        ctx.fill();
        ctx.closePath();
    }
    update() {
        this.x += this.velocity.x;
        this.y += this.velocity.y;
        this.radius -= 0.01;
        this.draw();
    }
}

// Felhasználói csúszka használata.
canvas.addEventListener("mousemove", (e) => {
    let rect = canvas.getBoundingClientRect().top;
    user.y = e.clientY - rect - user.height / 2;
});

canvas.addEventListener("touchmove", (e) => {
    let rect = canvas.getBoundingClientRect().top;
    user.y = e.changedTouches[0].clientY - rect - user.height / 2;
});

canvas.addEventListener("touchstart", (e) => {
    let rect = canvas.getBoundingClientRect().top;
    user.y = e.changedTouches[0].clientY - rect - user.height / 2;
});

// Ütközészlelés.
function collision(b, p) {
    p.top = p.y;
    p.bottom = p.y + p.height;
    p.left = p.x;
    p.right = p.x + p.width;

    b.top = b.y - b.radius;
    b.bottom = b.y + b.radius;
    b.left = b.x - b.radius;
    b.right = b.x + b.radius;

    return (
        p.left < b.right &&
        p.top < b.bottom &&
        p.right > b.left &&
        p.bottom > b.top
    );
}

// A labda visszaállítása
function resetBall() {
    ball.x = canvas.width / 2;
    ball.y = canvas.height / 2;
    ball.speed = 0;
    ball.velocity.y = 0;
    ball.velocity.x = 0;

    timeout = setTimeout(() => {
        ball.x = canvas.width / 2;
        ball.y = canvas.height / 2;
        ball.speed = 7;
        ball.velocity.y = Math.random() < 0.5 ? 7 : -7;
        ball.velocity.x = Math.random() < 0.5 ? 7 : -7;
        console.log("g");
    }, 1500);
}
```

```
// GameOver funkció
function gameOver() {
    let stat;
    let maxScore = 10;

    // These codes match with both players.
    function commonCodes() {
        ball.x = canvas.width / 2;
        ball.y = canvas.height / 2;
        ball.speed = 0;
        cancelAnimationFrame(gameId);
        panel.classList.add("reveal");
    }
    if (user.score >= maxScore) {
        stat = "Nyertél!";
        commonCodes();
    } else if (computer.score >= maxScore) {
        stat = "Vesztettél!";
        commonCodes();
    }
    statPanel.textContent = stat; // A statisztika megjelenik a statisztikai panelen
}

function draw() {
    drawRect(0, 0, canvas.width, canvas.height, "#089c29");
    drawRect(user.x, user.y, user.width, user.height, user.color);
    drawRect(
        computer.x,
        computer.y,
        computer.width,
        computer.height,
        computer.color
    );
    drawArc(ball.x, ball.y, ball.radius, ball.color);
    drawNet();
    drawText(user.score, canvas.width / 4, canvas.height / 5);
    drawText(computer.score, (3 * canvas.width) / 4, canvas.height / 5);
}

let gameId; // Az azonosító, amely befejezi és elindítja a játékot.
let timeout; // A labda intervallumának változója.
let particles = [];
function game() {
    gameId = requestAnimationFrame(game);
    draw();
    gameOver();

    // Ütközésérzékelés felső és alsó falhoz.
    if (
        ball.y + ball.radius + ball.velocity.y > canvas.height ||
        ball.y - ball.radius < 0
    ) {
        ball.velocity.y = -ball.velocity.y;
    }

    // Ütközésérzékelés bal és jobb falhoz.
    if (ball.x + ball.radius + ball.velocity.x > canvas.width) {
        // Ha a labda a jobb falat érinti, a felhasználó pontot kap.
        resetBall();
        userScore.play();
        console.log(userScore);
        user.score += 1;
    } else if (ball.x - ball.radius < 0) {
        // Ellenkező esetben, ha a labda a bal falat érinti, a számítógép pontot kap.
        resetBall();
        comScore.play();
```

```
        console.log(comScore);
        computer.score += 1;
    }

    // A labda helyzetének növelése.
    ball.x += ball.velocity.x;
    ball.y += ball.velocity.y;


    computer.y += (ball.y - (computer.y + computer.height / 2)) * compSpeed;

    // Melyik játékos fog most ütni.
    let player = ball.x + ball.radius < canvas.width / 2 ? user : computer;

    // Ha ütközés történik,
    if (collision(ball, player)) {

        let collidePoint = ball.y - (player.y + player.height / 2);
        collidePoint = collidePoint / (player.height / 2);
        let angle = (Math.PI / 4) * collidePoint;
        let direction = ball.x + ball.radius < canvas.width / 2 ? 1 : -1;
        ball.velocity.x = direction * Math.cos(angle) * ball.speed;
        ball.velocity.y = Math.sin(angle) * ball.speed;


        ball.speed += 0.5;

        // Részecske-robbanás
        if (player == user) {

            for (let i = 0; i < ball.radius; i++) {
                let x = player.x + player.width;
                let y = ball.y + ball.radius;
                particles.push(
                    new Particle(
                        x,
                        y,
                        `hsl(${Math.round(Math.random() * 360)}, 50%, 50%)`,
                        Math.random() * 3 + 0.5,
                        {
                            x: Math.random() * 3,
                            y: (Math.random() - 0.5) * 3,
                        }
                    )
                );
            }
        } else if (player == computer) {

            for (let i = 0; i < ball.radius; i++) {
                let x = player.x - player.width;
                let y = ball.y + ball.radius;
                particles.push(
                    new Particle(
                        x,
                        y,
                        `hsl(${Math.round(Math.random() * 360)}, 50%, 50%)`,
                        Math.random() * 3 + 0.5,
                        {
                            x: -Math.random() * 3,
                            y: (Math.random() - 0.5) * 3,
                        }
                    )
                );
            }
        }
    }
```

```
    particles.forEach((particle) => {

        if (particle.radius <= 0) {
            particles.splice(particle, 1);
        } else {
            particle.update();
        }
    });
}

play.addEventListener("click", () => {
    cancelAnimationFrame(gameId);
    user.score = 0;
    computer.score = 0;
    panel.classList.remove("reveal");
    resetBall();
    clearTimeout(timeout);
    ball.speed = 7;
    ball.velocity = {
        x: 7,
        y: 7,
    };
    if (mode.value == "") {
        alert("Please Select A Mode First!");
        panel.classList.add("reveal");
        return;
    }

    compSpeed = mode.value;

    game();
});

window.addEventListener("resize", () => {
    Resize();
});

function Resize() {

    if (window.innerWidth <= 620) {
        alert("Please Rotate Your Device.");
        canvas.height = window.innerHeight;
        canvas.width = window.innerWidth;
    } else {
        canvas.width = 600;
        canvas.height = 400;
    }
}

Resize();
```

2. Felhasználói dokumentáció

A Pong játék egy böngészőben futó alkalmazás. Letöltés után böngészőjében tudja megnyitni.

A játék alapvetően három ablakból tevődik össze, ezek a következők:

1. kezdőlap
2. Játéktér
3. Játék vége

2.1. Kezdőlap

A játék kezdőlapján lehetőség van kiválasztani, hogy milyen módban szeretne játszani. A módok között az jelenti a különbséget, hogy a számítógép játékos milyen gyorsan mozog az ütővel.

A játékmódok a következők:

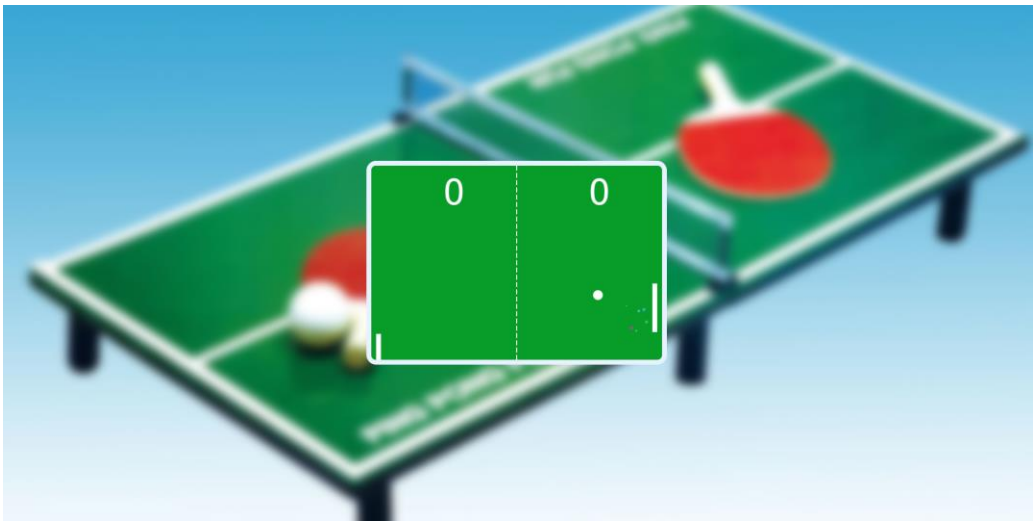
- Kezdő (baby) - Ez a leglassabb játékmód.
- Könnyű (easy) - Enyhén gyorsabb, mint a kezdő módban.
- Normál (normal) - Közepes sebességű játékmód.
- Nehéz (hard) - Gyorsabb, mint a normál mód.
- Profi (insane) - Ez a leggyorsabb játékmód.

Miután kiválasztotta a felhasználó a játékmódot, akkor a játék kezdete gombra kell kattintania.

2.ábra- Kezdőlap

2.2. Játéktér

A játéktéren a játékos a számítógép ellen játszik a kiválasztott játékmódban. Az ütőt a játékos az egér segítségével tudja mozgatni. A játék addig tart amíg 10 pontot nem gyűjt össze valamelyik játékos. A labda mozgását egy kis animáció követi, valamint, ha a labda a falhoz ér, akkor hangot ad ki. Minden pontszerzés után a labda a játéktér közepéről indul újra.



3.ábra - játéktér

2.3. Játék vége

Ha a pontszám elérte a 10 pontot, akkor a játék véget ér.

4.ábra – játék vége

Összefoglalás

A Pong egy klasszikus asztali teniszjátékot szimuláló videójáték, ahol két játékos egy-egy ütővel próbálja eltalálni a labdát, és megakadályozza azt, hogy a saját oldalára essen. Az egyik játékos a számítógép, míg a másik játékos az egérrel irányítja az ütőjét. A labda a játék elején középen helyezkedik el. A játékosok felváltva próbálják visszapattintani a labdát az ellenfél oldalára, elkerülve, hogy a saját oldalukon essen le. Ha a labda az ellenfél oldalára esik, akkor az adott játékos pontot szerez, és a labda visszakerül a középre. A játék addig folytatódik, amíg valamelyik játékos eléri a meghatározott pontszámot.

Továbbfejlesztési ötletek:

- Módosított játéksebesség: Különböző nehézségi szinteket vagy sebességi opciók, hogy a játékosok testre szabhassák a játékot a saját preferenciáik szerint.
- Power-upok: Vegyél fel különféle power-upokat a játékba, amelyek különleges képességeket adnak a játékosoknak. Például, egy power-up, ami megnöveli az ütő méretét, vagy egy power-up, ami megváltoztatja a labda sebességét.
- Többjátékos mód: több mint két játékos részvételére
- Kihívások és pályák: változatos kihívásokat és pályák a játékhoz. Például, olyan pályák, ahol akadályokat kell kikerülni, vagy olyan kihívások, ahol a labda különleges módon viselkedik (pl. gyorsabban repül, megváltozik az iránya stb.)
- Grafikai fejlesztések
- Játékstatisztikák: játékstatisztikákat megjelenítő rész, ahol a játékosok nyomon követhetik a pontszámukat, a győzelmeik számát vagy akár az átlagos reakcióidejüket. Ez motivációt adhat a játékosoknak a fejlődésre és versenyzésre.
- Játékmenet variációk: Például, speciális ütések vagy kombinációk, amelyek extra pontokat eredményeznek, vagy időlimit a fordulókra, ahol a játékosoknak minél gyorsabban kell pontot szerezniük.

Irodalomjegyzék

Webprogramozás és játékfejlesztés HTML5 és JavaScript segítségével" - Csík-Kovács Zoltán

Modern webfejlesztés HTML5 és CSS3 alapokon" - Bánáti Pál, Oláh István, Szabó Tamás

"Webfejlesztés HTML, CSS, JavaScript alapokon" - Tóth Bálint, Dénes Dávid, Dudás Gergő

W3Schools (https://www.w3schools.com/)